第1章

アーキテクチャの視点でみたARMコアの変遷と動向

――LSI設計者は「ファミリ」に，ソフト開発者は「アーキテクチャ」に注目

M.P.I

ARMプロセッサは歴史を積み重ねてきたRISC CPUだが，組み込み機器の中でのみ使われてきたこともあり，そのシリーズ展開は意外と知られていない．ここでは「ファミリ」と「アーキテクチャ」の観点で，ARMコアの変遷と動向について解説する．最新のCortexファミリの概要にも触れる．　（編集部）

すでに32ビット組み込みプロセッサ市場の王者となって久しいARMプロセッサですが，この先のさらなる発展のためか，現在はどうも転換のステージに入っているように見えます．そして新たな展開の尖兵となるのが，V7アーキテクチャ，および「Cortex」というネーミングのもと，「A」，「R」，「M」の3文字を冠した新ファミリです．

とはいえ，現在は，ARM7ファミリとARM9ファミリがその応用の広さでも数量でも圧倒的です．従来からの変遷をここで一度整理しておくことは，ARMプロセッサの理解には必須でしょう．なぜならARMプロセッサは非常に互換性を重んじて進化してきています．応用ごとの組み込み開発と割り切ってしまい，互換性など無視して最適化を図るようなプロセッサ開発手法が最近では注目を集めている中で，これまでのARMのポリシは非常に伝統的なものでした（図1）．

また，ARMファミリでは過去の世代のプロセッサも適材適所で使われ続けています．複数の世代が並行に応用され，共存しています．ARM11ベースの製品が出荷される一方で，いまだにARM7が現役として新たな製品に応用されているのです．新世代の登場とともに過去の世代が急速に廃(すた)れたり，ひどい場合にはIPベンダそのものが（新世代の拡販などのため）みずからの過去の世代の製品を積極的にフェーズアウトさせるようなケースもあると，時折聞きます．これに対して英国ARM社は，古いファミリに対する保守や設計手法の更新対応もきちんと継続してきています．その結果，現状，世代の異なる非常に多くの種類のコ

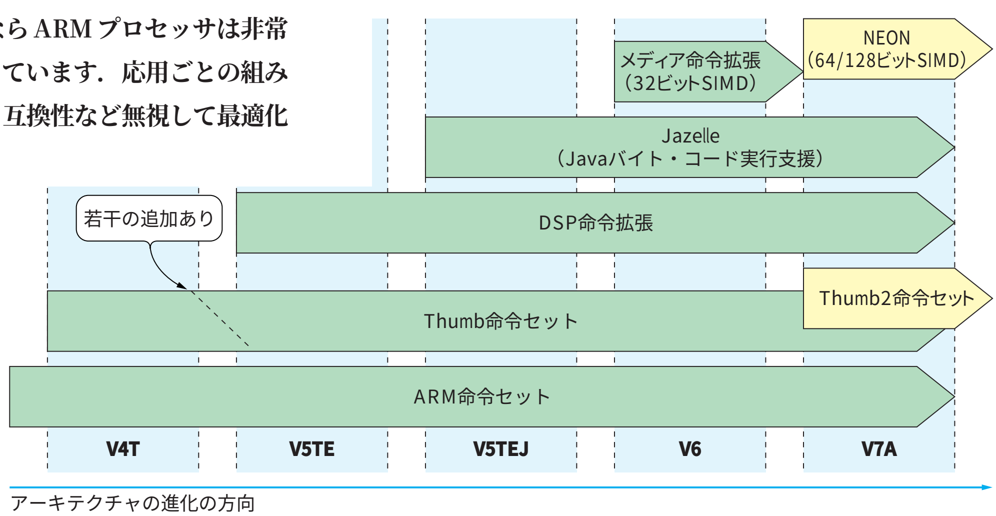


図1
ARMアーキテクチャの主要命令セットの変遷
バージョンが上がるにつれて，命令セットが互換性を保って拡張されていることがわかる．

KeyWord　ARM，RISC，CPU，Cortex，Thumb，MMU，SIMD，Jazelle，ARM7TDMI，ARM926EJ-S

アが並存し，用途に応じて使い分けられています(図2)．

本稿では，まず，ARM各ファミリに共通する特徴を簡単にまとめた後，現存の“ARM”という名称を冠する多様なコアの理解のために，「ファミリ」と「アーキテクチャ」と呼ぶARM特有の概念を説明します．なお，ここではARM7登場以降のARMアーキテクチャの変遷を扱うことにします．なぜなら，現在市場に存在する多様なARMベースの製品を理解しようとする場合，とりあえずARM7以降，ARM11までの話がわかっていれば十分だからです．ARM6までの歴史については，下掲のコラム「ARM7前史」に簡単にまとめておきます．また，これから登場してくるであろう新世代のCortexについても，まだ全ぼうは明らかではないものの触れておきます．

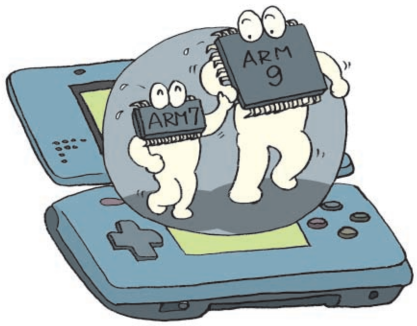


図2　ARMは複数の世代のコアが併存して使われている
現状，世代の異なる非常に多くの種類のコアが並存し，用途に応じて使い分けられている．

● 組み込み応用をねらった“濃い味付け”のRISC CPU

ARMプロセッサは，“ごく普通”の32ビットRISC(reduced instruction set computer)プロセッサです．ロード/ストア型アーキテクチャのレジスタ・マシンであり，フラットでリニアなメモリ空間にアクセスし，命令セットは固定長です．しかし，RISCプロセッサとしては濃い味付けがなされたアーキテクチャでもあります．これは基本となるARM7TDMIから変わっていません．以下に，ARMプロセッサの独特の部分を列挙します．

1）基本は3オペランドで32ビット固定長の“ARM”命令セットだが，現行のARM応用製品のほぼすべてで16ビット固定長の“Thumb”命令セットが併用されている

2）“ARM”命令セットには，全命令に条件フィールドがあり，条件付き実行が可能である．これにより，分岐によるパイプラインの乱れを最小化できる．また，ほとんどの命令でオペランドをビット・シフトできる．1命令当たりの可能な操作が大きい例外的なRISCプロセッサといえる．反面，ほかのRISCプロセッサで一般的な遅延分岐は排除している

3）レジスタ・バンクに対応している

4）コプロセッサ命令による命令セット拡張に対応している

1)は，組み込み応用で厳しいメモリ容量の制限に配慮し，プログラム容量の削減に貢献するのが目的です．**3)**は，リアルタイム制御などを考慮した割り込みに対するレスポンスの改善がねらいです．**2)**はハンド・コーディングでアセンブラによりミドルウェアなどをガリガリとチューニン

1

Column　ARM7前史

ARMというプロセッサは，ARM社という会社ができる以前から存在していました．その開発は1980年代前半の英国Acorn Computers社に行き着きます．ほかのCPUアーキテクチャの源流と同じように，それはデータ処理のためのCPUとしての出発であったようです．

実際，Acorn社がARMベースのコンピュータを世に出したのは1980年代の中盤でした．当時，パソコンでもワークステーションでも，すでにほかのCPUアーキテクチャが市場を独占していたことはご存じのとおりです．そのままであれば，ARMプロセッサの発展は英国の片隅で終わっていたことでしょう．

ARMプロセッサにとって幸運だったのは，米国Apple Computer社が新概念の携帯型端末“PDA(Personal Digital Assistants)”として登場させた「Newton」のCPUにARMプロセッサを採用し，そのためにAcorn社とは別のARM社というCPU開発専業会社が設立されたことです．さらに幸運なことにARM社は，最初から設計中心のファブレスIPベンダとしてスタートできました．Acorn社，Apple社に続く第3の出資者は，今はなくなってしまった米国VLSI Technologies社であり，チップはそこで製造されることになっていたのです．

この1990年代初頭〜中盤の段階で，ようやく業界内でARM6，そしてARM7という名のプロセッサが認知され始めました．これが世間に広く知られた最初のARMプロセッサでした．しかしARMは，このときすぐに成功を収めたわけではありません．Newtonを始め，この世代のほかの応用は軒並み立ち上げに失敗してしまいました．

この苦境を救い，ARMプロセッサを今日の強力なポジションに導いたのは，超ベストセラーとなったARM7TDMIと，それをベースバンド処理のエンジンに採用した欧州の携帯電話メーカでした．そして欧州の携帯電話が世界市場で伸びるに従って，ARMプロセッサは携帯電話を中心とする組み込み応用の世界を制覇していったのです．

グしていく作業で威力を発揮する特性と思われ，アーキテクチャの源流の歴史を感じさせます．**4)** は，IPとして特定用途向けシステムLSIなどに組み込むときに，LSI設計者に拡張の余地を与えるものです．

もちろん，ARMプロセッサにも課題はあります．現存のほとんどの製品が対応しているThumb命令セット単独では割り込みハンドラなどは記述できず，つねにARM命令セットへの切り替えが行われます．その結果，ルーチンにより2種類のコンテキストのコードを使い分ける必要があります(ただし，最近のThumb2命令セットに対応するものからは，Thumb2命令，ARM命令をコンテキストで切り替える必要がなくなった)．また，**2)** の特徴から，演算回路の実装上，ほかのプロセッサと比べてクリティカル・パスが長くなりがちです．論理合成可能なソフト・マクロとして組み込まれる傾向と合わせて，クロック周波数を上げづらいなどの弱い面も持ち合わせています．

●「ファミリ」は設計のベースにしているコアを示す

まずは，ARMにおける「アーキテクチャ」と「ファミリ」について説明します(**図3**)．

いわゆるARM7，ARM9といった各グループを，ARM社では「ファミリ」と呼んでいます．ARM社の公式見解ではありませんし，一部例外もあるのですが，ざっくりと言ってしまえば，「ファミリ」は設計のベースにしているコア・プロセッサ(CPUコア)の違いで区別されます．基本性能とプロセッサの規模の目安になります．プロセッサ・パイプラインの段数は，数字の多いほうがより長くなります．より高速に動作できる代わりに，回路規模は大きくなると考えて，ほぼまちがいありません．LSI設計者にとっては，後述するアーキテクチャよりも，ファミリの概念のほうが重要でしょう．

なお，ARM7，ARM9といったものを「世代」と理解すると誤解を招きます．じつはARM9系のコアより後に設計されたARM7系のコアもあります．その実装方法もハード・マクロから論理合成可能なソフト・マクロへ(**図4**)，そしてデバッグ機能を充実させるなど，それぞれのファミリごとに何世代か変更を経てきています．とくに歴史のあるARM7の場合は，実装方法が大幅に異なるものが複数共

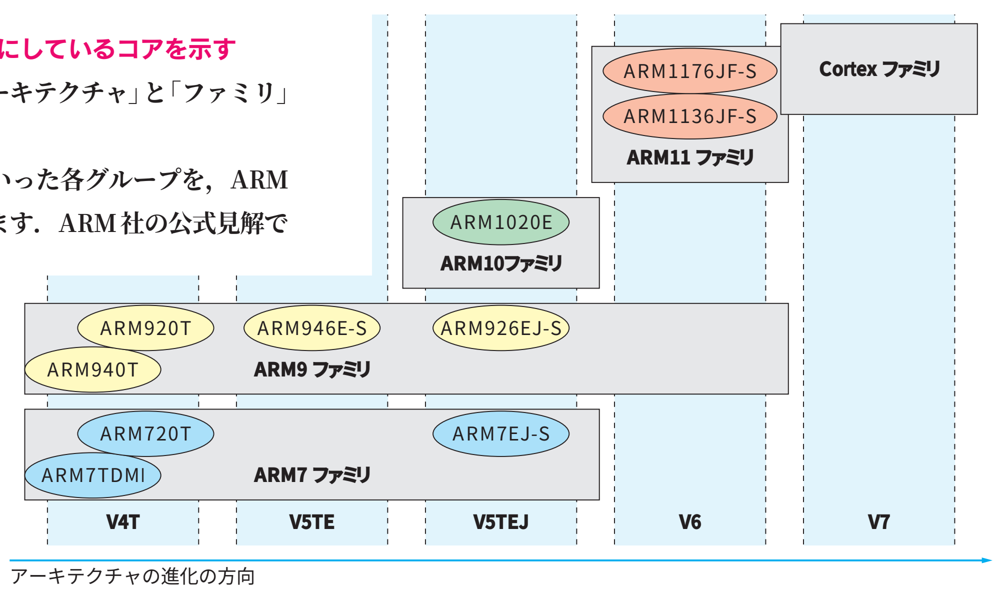


図3
ファミリとアーキテクチャの関係
同じファミリでも異なるアーキテクチャ(命令セット)を持つものが存在している．

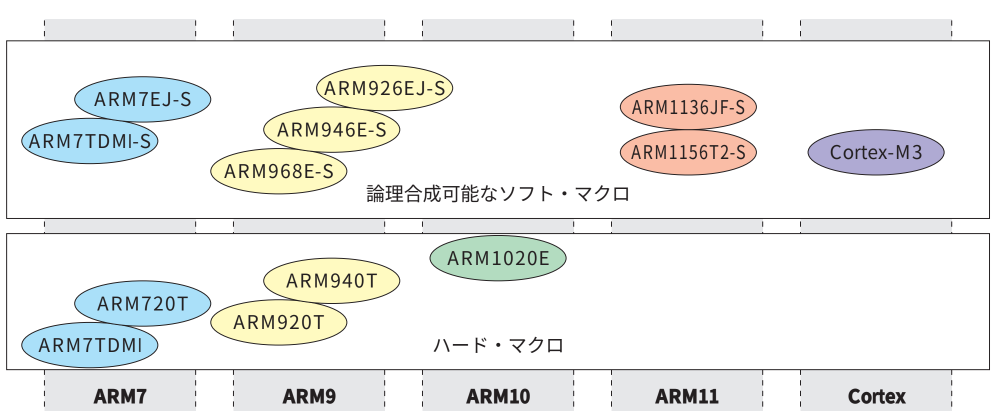


図4
実装による分類
現在の主力は論理合成可能なコアだが，ハード・マクロも依然として残っている．Cortexについては筆者の推定．将来のコンフィギャラブル，リコンフィギャラブル展開はあるのか？

存しています．機能的にもJavaアクセラレータ(Jazelle)付きのARM7もあれば，付いていないARM9もある，というぐあいで，一概にARM9がARM7より機能が上とは言えません．

● ファミリごとの特徴を理解する

以下に，ARM7以降のファミリの変遷をまとめます．

1）ARM7ファミリ

ARM7TDMIを中心とするファミリです．論理合成可能バージョンのARM7TDMI-Sもあり，ユニファイド・キャッシュ(命令キャッシュとデータ・キャッシュを単一にしたもの)を搭載したARM720Tもあります．携帯電話のベースバンド処理向けでブレークしました．軽くて使いやすいコアであるため，それほど絶対性能を必要としない組み込み機器に最適で，幅広い用途に数多く使用され，現在も現役であり続けています．一部の例外を除き，3段パイプラインのコアで，最先端のプロセスで製造しても，動作クロックはまず100MHzには届きません．

2）ARM8ファミリ

ARM社の製品にはARM8を冠したコアは，現在ラインナップされていません．旧DEC(Digital Equipment Corp.)のStrongARMがARM8に該当します．DECの半導体部門は米国Intel社に買収されたため，同社のXScaleがARM8の末裔と言えなくもありません．Intel社のARMは独自の進化を遂げていますが，アーキテクチャ的にはARM社のものと整合性をとるように配慮されています．

3）ARM9ファミリ

携帯電話のアプリケーション・プロセッサやPDAなどで，現在よく使われているのがARM9です．「DSP拡張」と呼ばれるMAC(multiply and accumulate)演算処理やJavaコードの実行支援機能を搭載したアプリケーション・プロセッサに分類されるものが多いようです．GUI(graphical user interface)機能を搭載したOSを載せ，BREW(Binary Runtime Environment for Wireless；米国Qualcomm社の携帯電話用ソフトウェア実行環境)やJavaなどを走らせ，画像や映像を再生するようなケースでは，ARM926EJ-Sなどが主力となっています．製造プロセスにもよりますが，5段パイプラインで高速をねらって設計しているケースでは，動作クロックが200MHz程度のものがあります．また，ARM9はもっとも品種の多いファミリで，キャッシュやJavaなどへの対応がない組み込み応用向けのコアも複数存在します．

4）ARM10ファミリ

整数演算プロセッサは6段パイプライン，ベクタ浮動小数点プロセッサは演算が7段，ロード/ストアが5段という，当時のARMにしては相当に高い性能をねらっています．ARM10コアのライセンスを受けた会社は，高速な浮動小数点演算など，ARM9では不可能な性能をどうしてもARM9のリリースと同時期に必要であったところに限られるのではないかと思われ，応用製品の事例はあまり聞きません．また，初期のARM10には論理合成可能なバージョンがなく，ARM10のリリースとはそれほど時間をおかずに論理合成可能なARM11が出たため，ARM10系は今後も主力にはならないと推測されます．

5）ARM11ファミリ

スカラ・プロセッサとしてはARM最高峰となるファミリです．画像，映像のエンコード処理など，ARM9では処理能力が足りないケースで主力として使用されるものと思われます．このファミリから，32ビット幅ながらSIMD(single instruction multiple data)命令が追加されました．メディア拡張命令と呼ばれているとおり，音声や画像の処理によっては高速化できるケースもありますが，ビット幅が狭いため，アルゴリズムしだいとなりそうです．浮動小数点の対応については，ARM10と同じベクタ浮動小数点プロセッサをオプションとして搭載できます．

6）Cortexファミリ

2005年から発表の始まった新ファミリです．Cortexという一つのネーミングを冠していますが，じつはまったくレベルの違うCortex-A，Cortex-R，Cortex-Mの3種類から構成されます．

Cortex-A8はARM11より上位にあるARM社初のスーパスカラ・プロセッサです．600MHz～1GHz程度の速度をターゲットにしているようです．Cortex-M3はシンプルで小さな組み込みコアです．ARM7に相当する，数十MHz～百数十MHz程度の動作周波数の組み込み応用のローエンド領域をカバーするものと思われます．Cortex-Rについては，公開情報がまだあまりありません．Cortex-Aより遅く，数百MHz程度までの条件の厳しいリアルタイム制御向けと考えられます．個人的にはCortex-Aのスカラ版カットダウン(機能や性能を抑えたローエンド版)のようなものになっていくのではないかと推測しています．

3種類のファミリは，V7アーキテクチャの採用によりThumb2対応の命令セットに一新されました．路線的には



それぞれ異なり，Cortex-AはARM11のさらに上位互換の延長（ARM12的存在）と位置付けられそうです．一方，ローエンドのCortex-Mについては「Thumb2命令のみ」という割り切りで，ARMプロセッサにしては珍しいソフトウェア互換性の断絶が存在します．

●「アーキテクチャ」は命令セットや機能の違いを示す

「ファミリ」に対して「アーキテクチャ」のほうは，命令セットを中心とする「機能」を定義したものと理解するべきだと思います．例えば，同じARM7ファミリでも異なるアーキテクチャのプロセッサがありえるということです．プログラマ（ソフトウェア開発者）のみなさんにとっては，ファミリよりこちらの概念のほうが重要でしょう．

アーキテクチャは，時代を経るにつれて機能が増える方向にあり，機能の多いほうがバージョン番号が大きいと思ってまずまちがいありません．ただし，付加的な機能についてはすべてのコアがサポートするわけではないので，機能の有無を仕様書などできちんと確かめるようにしてください．

アーキテクチャにはオプション機能，例えばTrustZoneといったセキュリティ対応の有無による分類もあります．煩雑になるので，こうした分類の説明はここでは割愛します．

● アーキテクチャごとの特徴を理解する

最近のアーキテクチャについて，簡単に説明します．

1）V4Tアーキテクチャ

ARMの代表品種であるARM7TDMIやARM9の第1世代であるARM920などがこれにあたります．ARM命令に加えて，16ビット固定長のThumb命令をサポートしています．これは，現行のARMのほぼすべての製品の基本となるアーキテクチャです．ほとんどのARM7系がこのアーキテクチャであり，何十億個かわかりませんが，現在までに非常に大量のチップが出荷されているものと思われます．

2）V5TEアーキテクチャ

V4TにMAC演算や飽和（saturation）演算に対応したDSP拡張を追加した，V5の基本となるアーキテクチャです．ARM9ファミリのARM946E-SやARM10ファミリのARM1020Eがこれに相当します．このアーキテクチャを採用した製品は，V4Tベースを上回るほどには多くないと推定されます．しかし今後，ARM9系の組み込み向けコアであるARM966/968などの採用とともに，出荷数が増えると思われます．

3）V5TEJアーキテクチャ

V5TEJは，V5TEにARM社のJavaバイト・マシンの実行支援機能であるJazelleを追加したものです．Jazelleでは，一部のバイト・コードをハードウェアで，複雑なものはソフトウェアで処理してJavaの実行効率を上げます．ARM9ファミリを代表する人気品種のARM926EJ-SなどがV5TEJを採用しています．V5TEの派生アーキテクチャですが，携帯電話のアプリケーション・プロセッサとして定番になっているので，ここでは独立して取り上げました．ARM7EJ-Sという型名のARM7ファミリ品も発表されていますが，じつはこれはARM7系コアではなく，ARM9系コアのカットダウンではないかと思われます．

4）V6アーキテクチャ

V5TEJに前述のメディア拡張機能を追加したものがV6アーキテクチャです．これは，32ビット幅を16ビット・オペランド×2個，もしくは8ビット・オペランド×4個とみなすSIMD拡張です．ARM11ファミリに適用されました．じつはこのV6の段階でThumb2命令セットも登場しているのですが，この時点ではあくまでもオプション機能の扱いでした．V6の段階では，従来のARM命令セット＋Thumb命令セットの混在使用が主流となっています．

5）V7アーキテクチャ

V7は，Cortexファミリで登場した新世代のアーキテクチャです．Thumb2命令の採用がV7アーキテクチャの共通部分です．Thumb2の採用により，従来わずらわしかったARM命令とThumb命令をコンテキストで切り替える手法と決別できました．ただし，V7対応であるCortexファミリの中でも，従来のARM（Thumb）命令セットの使用も可能でアーキテクチャ面の継承性の高いハイエンドのCortex-Aと，おそらく軽量化のためにThumb2に絞ってしまい，従来とは断絶のあるCortex-Mでは，ほとんど別のアーキテクチャと言うべきではないかと思います（ARM社はV7A，V7Mと呼んでいるが，その差は大きい）．

また，Cortex-A系では，NEONという名で128ビット幅のSIMD命令をサポートするようになり，演算能力は格段に進歩しています（ARM11ではSIMDといいながら，32ビット幅にとどまっていた）．

● 内蔵機能を型名から推定できる

ARMプロセッサの構成は，キャッシュの有無やMMU

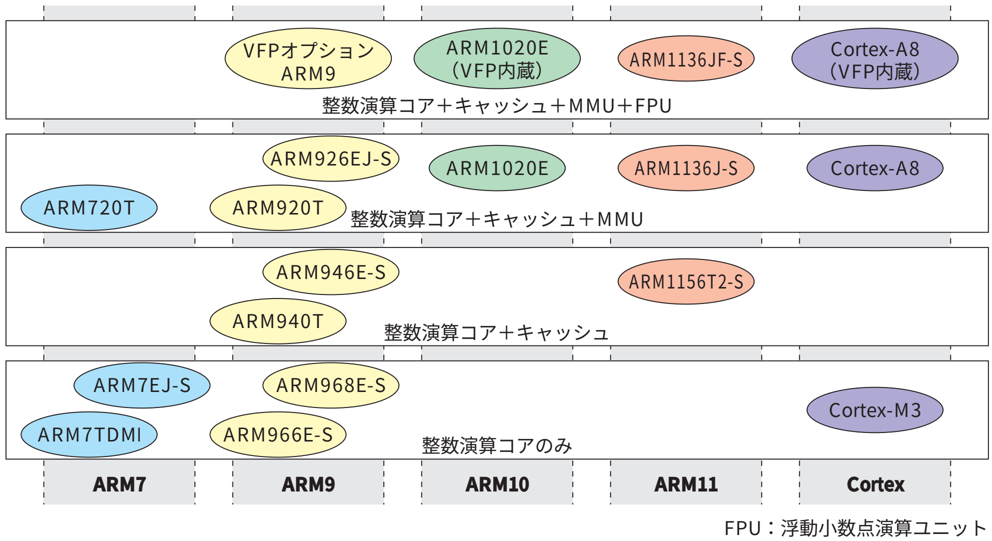


図5　キャッシュ，MMU，FPUの有無とファミリ内の製品
同じファミリでも，キャッシュ，MMU，FPU（浮動小数点演算ユニット）との組み合わせにより多様なコアがある．

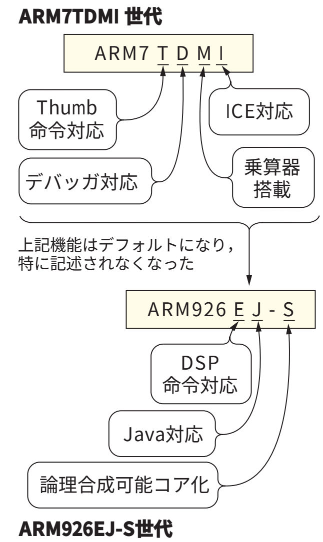


図6　代表的な型名と機能
機能の記述方法がリリース時期とともに変わってきている．

（memory management unit）の有無でも分類することができます．こちらは**図5**に概要を示します．ARM社特有の事情ですが，キャッシュ，MMU付きは10の位が「2」，MMUなしでキャッシュ付きは10の位が「4」，キャッシュ，MMUともなしは10の位が「6」などとなっています．型名の10の位については，数字の小さいもののほうが上位になっているので注意してください．

一般論で言えば，種々のアプリケーション・ソフトウェアを走らせる必要のあるOS（例えばLinuxなど）を搭載するのであれば，MMUが必須になることが多いようですが，実メモリだけ（MMUなし）で動作することを特徴としているOS実装もあり，一概には言えません．また，高速動作をねらうのであればキャッシュを搭載することが一般的ですが，キャッシュ・ミスによってリアルタイム性を厳しく制御することが難しくなるケースもあります．そのため，高速性と厳しいリアルタイム性が要求されるケースでは，あえてキャッシュを積まず，キャッシュレスの高速なメイン・メモリのみで構成したほうがよいこともあります．

また，**図6**に英字によるサフィックス表記の例を挙げておきます．**図1**と**図3**～**図6**が頭に入っていれば，ARMコアの型名を聞いて，どのような素性のものか，だいたいのあたりをつけることができると思います．

いずれにせよ，ARMプロセッサの場合，選択の余地はありすぎるほどあるので，アプリケーションの特徴を見て選択するべきでしょう．

● オンチップ・バスのAMBAについても要確認

今回触れなかった部分では，ARM社のオンチップ・バスである「AMBA（Advanced Microcontroller Bus Architecture）」の知識が重要です．ARMコアに限らず，他社のIPコアでもAMBA対応のものが増えており，IPコアを利用したLSI設計を行う場合には必須の知識になっています．AMBAそのものも何代かの世代交代を経てきており，これまた複数の世代が並存しているので，よく確認してください．また，OptimoDE（VLIW方式を採るDSPコア）という当面ARMコアとは別系統の新技術も今後のARMのアーキテクチャに影響を与えるかもしれません．

ARMプロセッサは，現在も進化しながら急速に発展しているプロセッサなので，その全貌を短いページにまとめるのは容易ではありません．駆け足になってしまったことをお詫びさせていただきます．

M.P.I
マイコン開発マネージャ

<筆者プロフィール>
M.P.I（ペンネーム）．若いころ，米国系の半導体会社で8ビット，16ビットのプロセッサ設計に従事．ベンチャ企業に移って，コードはコンパチ，ハードは独自の32ビット互換プロセッサのアーキテクトに．米国，台湾の手先にもなったが，このごろは日本の半導体会社でRISCプロセッサ担当の中間管理職のオヤジ．

1